**转码工具**

基于cs2lua，快速Unity c#代码转换为lua脚本。

1、修改bug，提交c#代码。

2、指定Assembly-CSharp.csproj。

3、转换为lua脚本。

4、打包发布。

**目录**

/bin/cs2lua 转码工具

/bin/Mix.py 签名混淆脚本

/src 转码脚本

/out 热更脚本

/\_signed 签名脚本

**流程 (转码->签名)**

\*.c#

|

cs2lua.exe

|

handler.py

|

Sign.bat

|

\*.lua

**1.Cs2lua.exe**

1、修改Run.bat，指定工程Assembly-CSharp.csproj。

2、Run.bat –gen

----> 工程目录/lua/\*.lua

**2.handler.py**

cs2lua出来的脚本存在很多cs2lua特定的结构和语句，但目前我们不全部沿用cs2lua框架，所以通过该脚本提取部分内容，并且对特定结构和语句进行转换，使风格更接近xlua。

最终出来的脚本，建议要对照一次c#原有函数逻辑，因为处理不保证完全正确，可能会出现语法、字符上的错误，或新规则没有匹配出来,还存在cs2lua相关结构语句。

步骤：

1. cs2lua生成的脚本

----> ./ src/\*.lua

1. Run.bat –xlua

----> ./out/\*.lua

1. Run.bat –copy ( Sign.Bat )

----> ./\_signed/\*.lua

脚本目录结构:

/out/core/ : 框架、接口相关的脚本。

/out/game/：具体的热更模块。（每次变动的部分）

**3.签名**

1. Sign.bat

----> ./\_signed/\*.lua。

带混淆的签名:

1.Sign.bat [-p] [-m]

-p 转为字节码，x86/x64

-m 开启文本混，0/1

**4.打包发布**

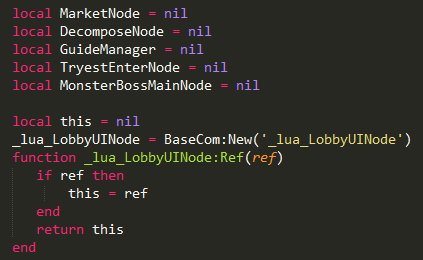
1. 打包./ \_signed /\*.lua。

2. 拷贝到svn://192.168.2.106/cehua/cehua/美术资源/动态更新图片资源，发布。

**热更脚本**

转码出来的热更脚本，即一个脚本对应一个c#文件，其中每个脚本包括以下几个结构，以” LobbyUINode.lua”为例：

**头部 & 持有对象**

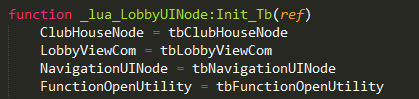


1. 模块名：\_lua\_LobbyUINode Lua对象：self

2、C#引用：由\_lua\_LobbyUINode:Ref(ref)获取，保存为this。

后续操作里，使用This.XXX、this:XXX()进行调用。

**C#类型声明**



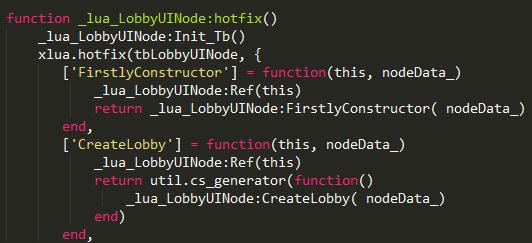
为使用方便，对部分CS类型提供省略名， 全部定义在/core/CSConfig.lua， Main.lua初始化时，映射为全局变量并进行xlua.private\_accessible处理。

xlua.private\_accessible：

XLua默认只能访问public类型，

对于私有对象，需要对类进行private\_accessible。

**热更函数**



1、热更函数，在Main.lua初始化时执行。

2、存在return util.cs\_generator的为Ienumerator方法。

其他为普通方法。

3、调用，如果该方法存在hotfix，则调用hotfix的方法，否则调用c#层原有方法。

**4、剩下的部分为具体的方法实现。**